Emotion Classification in Arabic Poetry using Machine Learning

Ouais Alsharif
Damascus University,
Department of Artificial Intelligence
Damascus, Syria

DeemaAlshamaa
Damascus University,
Department of Artificial Intelligence
Damascus, Syria

Nada Ghneim
Higher Institute for Applied Science and Technology,
Department of Informatics
Damascus, Syria

ABSTRACT
In recent years, work on sentiment analysis and automatic text classification in Arabic has seen some progress. However, the problem of emotion classification remains widely under-researched. This work attempts to remedy the situation by considering the problem of classifying documents by their overall sentiment into four affect categories that are present in Arabic poetry- Retha, Ghazal, Fakhr and Heja. This work begins by building an emotional annotated Arabic poetry corpus. The impact of different levels of language preprocessing settings, feature vector dimensions and machine learning algorithms is, then, investigated and evaluated on the emotion classification task.

General Terms
Natural language processing, classification, machine learning, sentiment analysis

Keywords
Arabic, poetry, emotion

1. INTRODUCTION
In everyday human-human interaction, emotional cues play a fundamental role in our perception and understanding of the outside world. Furthermore, as human-computer interaction experiments suggest, emotions come into play when we deal with technology as well. Reference [1] showed that people interact on a social and affective level with their computers the same way they interact with other people. Reference [2] also suggested that the exhibition of emotions is fundamental in achieving believability of agents. The importance of creating intelligent user interfaces that are social and affective was recognized by Picard [3] who called this field of research "Affective Computing". Researchers in this area have studied several ways for detecting user's emotions through speech [4][5], bio-signals [6], text [7] and others.

Due to the fact that most User Interfaces (UIs) today are textually based, research focused on text affect detection can play a significant role in improving current UIs and making them more socially aware. Several approaches have been undertaken for emotion detection in English including machine learning algorithms [8][7][9], keyword spotting, lexical affinity, hand-crafted models [10] and symbolic methods [11]. In Arabic, most work has focused on text classification based on topic rather than on affect [12]. Also, little work exists for sentiment analysis in the Arabic language [13][14]. However, as far as we are aware, no published workexists for emotion classification in Arabic, and the only work that could be classified under this category is that of [12].

This paper attempts to partially fill the gap in this area of Arabic NLP by addressing the problem of emotion classification using a corpus of Arabic poems written in the standard Arabic language. Since Arabic UIs are written in standard Arabic, and not in the numerous dialects used in informal communication usually used on blogs and websites, the choice of an Arabic corpus was limited. This issue, as well as other issues mentioned in section three were reasons why a poetry corpus was specifically chosen for this work. To achieve emotion classification, we used a machine learning approach and used different combinations of preprocessing settings, feature vectors and algorithms.

The rest of this paper is organized as follows. Related work is reviewed in section two. In section three, we give background information on the Arabic poetry domain and the corpus this work deals with. An outline of the approach is presented in section four, and in section five, the preprocessing methods are explained. In section six, feature selection is presented, then the different machine learning methods used are discussed in section seven.Section eight,presents the final results, with a conclusion in section nine.

2. RELATED WORKS
A fair amount of approaches have been utilized for affect classification in English. The most naïve of which is Keyword Spotting where emotion categorization occurs based on the detection of unambiguous affect words like "sad" and "happy". Reference [15]'s Affective Lexicon and [16]'s Affective Reasoner are two examples for this approach.

Another approach is Lexical Affinity which assigns a probabilistic affinity for particular emotions to arbitrary words. This method usually outperforms Keyword Spotting. Nevertheless, it's not suitable for developing a domain-independent model because its probabilities are often biased according to the domain of the linguistic corpora used to generate them [10].

Other methods require a deep understanding of the underlying semantics of language such as the work proposed by [10] and which utilizes a generic "common-sense" knowledge-base and four linguistic models to classify the emotion of the text at the sentence level. Another method suggested by [17] rests on psychological theories about human goals, desires and needs to build hand-crafted models of emotion.

Machine learning algorithms have also been used in textual affect sensing. Those systems can take into account not just the emotion keywords, but also probabilistic affinity, word co-occurrence frequencies and punctuation [10]. Works that have treated emotion and mood classification as a text-categorization problem used either flat [7][18] or hierarchical classification [9]. Reference [9] used hierarchical text categorization to specify the mood of writers' in online blogs. This resulted in an improved accuracy over the results

produced by [7]'s flat classification on the same corpus and using the same machine learning algorithm (SVM) [9].

In Arabic, the goal of [12] was to evaluate the performance of SVMs and C5.0 in the application of Arabic text categorization. One of the corpora used was a poetry corpus and the best accuracy for that corpus "49.15%" was achieved using C5.0.

2. THE ARABIC POETRY DOMAIN

This work is mainly concerned with the Arabic poetry domain.The choice of domain was due to its relative convenience comparing to other domains, as Arabic poems tend to be concise in the utilization of Arabic grammar and spelling unlike various other web based resources.

Moreover, the availability of free online repositories containing hundreds of poems allowed a rapid construction of the body of texts on which this work is built.

However, despite the convenience of the well-formed Arabic found in old poems, this domain presents a bigger challenge from an emotion classification point of view than do other domains such as movie reviews, blog posts and news' articles. The challenge is depicted by the convoluted presence of emotions throughout the body of a poem. Unlike poets of the modern days, old poets were inclined not to display emotion explicitly, but rather, through complex, convolved structures. These complex structures pose a hurdle to classification methods trained on language tokens since they don't take into account higher level language structures such as semantics and pragmatics where most of the emotional information lies. This is probably one of the reasons that Arabic poetry was the category with the least classification precision in [12].

The corpus was built using a small web scraping application to gather Arabic poems and their associated affect class from an online database of Arabic poems:http://www.toarab.ws. In total, we downloaded 1231 Arabic poems that vary in length in four main affect categories which are: Retha, Ghazal, Fakhr, Heja. The downloaded poems were preprocessed such that they were clean from scraping errors, diacritics and stop-words. Also, the two-versed poems were merged into single versed to compensate for the fact that each two verses constitute a linguistic unit.Table I shows statistics about our corpus.

3. METHOD OUTLINE

The pipeline of the work is as follows: first, the poems are cleansed from HTML residues. Diacritics and stop-words are removed, then two versed poems are transformed into one versed poems. After this initial cleaning the resulting poem body is either stemmed, rooted or left as is. Afterwards, the words were either kept by their POS tag as in [19] or left untouched. Afterwards, using a bag of words approach, a feature vector was constructed in two different ways. Then, different machine learning algorithms were applied on the resulting body of vectors to estimate the classification quality.

4. PREPROCESSING METHODS

Preprocessing is usually of minor importance in classification problems compared to feature selection and application of machine learning algorithms. However, in the context of

Class	Poem Count	Single-versed Poems	Two-versed Poems	Verse count	Average
Fakhr	693	3	690	14841	21.4
Heja	290	5	285	2304	7.9
Ghazal	144	0	144	1142	7.9
Retha	104	4	100	1754	16.9
Total	1231	12	1219	20041	16.4

Table 1.Corpus Statistics

Arabic poems, preprocessing seems to hold the key for better classification rates as most Arabic poems are built on the so called seas of poetry "بحور الشعر", which are meters for the rhythm of a certain poem. The rhythm of a poem, reflected by its seas, offers insights into the nature of the poem itself and the emotions held by the poet. In this work, however, we do not investigate the utility of poetry seas any further.

The rich morphological nature of Arabic requires some sort of preprocessing to be performed on words to return them to a normal form. Without such preprocessing, the algorithms applied later in the pipeline could mistake two very similar words as different such as: "قاتل" and "قتّال". The tools available at hand to compensate for such differences are stemming and rooting.

Often confused in the literature, stemming is the procedure of returning the words to their stems through removing prefixes and suffixes Whereas rooting is returning the words to their roots. The two are confused probably due to the inexistence of word stems in Latin languages. In Arabic, however, the difference is paramount. For example the word: "المحقق" has a stem "محقق" and a root "حقق". Note how the stem preservers the word structure and morphological pattern, whereas the root doesn't.

This work tests two different software tools for preprocessing the poems, Alkhalil morphological analyzer [20] and Khoja stemmer [21]. Despite its name however, Khoja does not perform stemming but rather rooting. Alkhalil does both rooting and stemming. Of course, on eachexecuted test, rooting, stemming or neither was performed.

5. FEATURE SELECTION

Constructing the feature vector is a crucial part in any machine learning application as it pertains to how each object is represented through its features. The problem is defined as follows: let Dbe a set of documents $D = d1\ d2\ ...dn$ where each document is represented by a sequence of words $d = w1\ w2\ ...wm$ and let $g$be a function that maps from the domain of documents to the domain of features $\vec{d} = g(d)$. The function g could be a linear or a nonlinear function. In this work we use two nonlinear functions: The first function returns the top k unigrams with most occurrences throughout a document set D for every class and then concatenates the resulting vectors. The second function, called a mutual deduction function selects the top k unigrams with the most mutually deducted occurrences throughout a document set D and also concatenates the resulting vectors. Both functions result in a feature vector of length: $C * k$ where C is the number of classes and k is the number of features in each class. The mutually deducted occurrence is defined as follows: let $n_c(f_i)$ be the number of occurrences for a

potential feature f_i in class c, then the mutually deducted count for feature $f_i$in class c is as follows:

$$dn_c(f_i) = n_c(f_i) - n_d(f_i), \quad where\ d \neq c$$

Which means that the number of occurrences of feature f in class c is discounted from the number of the occurrences of the same feature in all other classes.

To construct the representation of a document D once the feature vector is formed, simply iterate over the feature set and place a Boolean flag when a feature is found. The Boolean vector model was chosen due to its outperformance to the count model in the closely related problem of sentiment analysis [22].

6. MACHINE LEARNING METHODS

This work aims to test if the problem of emotion classification in Arabic is susceptible to the tools used for text categorization and emotion classification in English.

Four algorithms are tested: Naïve Bayes, Support Vector Machines, Hyperpipes and Voting Feature Intervals. To utilize these algorithms the standard bag of words approach is used on the features $\{f1, f2, \dots, fn\}$ that result from the previous step.

6.1 Naïve Bayes

The theory behind the Naïve Bayes algorithm relies on assigning a class to a document according to the following equation:

$$c^* = argmax_c P(c|d) \quad (1)$$

Where c* is the assigned class, c is class and d is a document.

$$c^* = argmax_c P(d|c) * \frac{P(c)}{P(d)} (2)$$

And since P(d) plays no role, the equation becomes:

$$c^* = argmax_c P(d|c) * P(c) (3)$$

The Naïve Bayes algorithm then makes the assumption that each feature is conditionally independent of other features given a document resulting in the following equation:

$$c^* = argmax_c \prod_1^n P(f_i|c) * P(c) (4)$$

Despite the fact that the assumption made by the Naïve Bayes algorithm does not generally hold, the algorithm performs well in practice [23]. It also has been utilized successfully in the closely related English Movie Review sentiment classification works, as in [22] and [19].

6.2 Support Vector Machines

A support vector machine (SVM) is a non-probabilistic linear classifier. Unlike Naïve Bayes, SVMs take a geometric optimization approach to the classification problem, seeking to construct a hyperplane or a set of hyperplanes in a high dimensional space. The algorithm relies on the intuition that the generalization error is minimized by maximizing the geometric distance between the separating set of hyperplanes and the closest training data point. This formulation gives SVMs some neat properties such as their immunity to local minima (as compared to Neural Networks).

SVMs are also universal learners, that is, by using an appropriate kernel, SVMs can be extended beyond linear classification to learn polynomial classifiers. There are several reasons for experimenting with SVMs for the task at hand. Primarily, [24] lists four main points on why SVMs would work for a text categorization problem: high dimensional input space, few irrelevant features, sparseness of document vectors and the susceptibility of text categorization problems to linear separation.

Experimental results on SVMs for text categorization show significant improvements over other classifiers. The work in [22] attributes the highest classification accuracy to SVMs (compared with Naïve Bayes and Maximum Entropy). In [25] also, Yang and Liu, rank SVMs as the best classification algorithm compared to k-nearest neighbor, Linear Least Squares Fit, Naïve Bayes and Neural Networks. SVMs were also shown to perform well in previous studies on mood classification [9].

6.3 Voting Feature Intervals

Voting Feature Intervals (VFI) is a non-incremental learning algorithm. Proposed initially by Demir**ö**z and Güvenir[26], VFI is far underutilized in the literature as compared to Naïve Bayes and SVMs. (Microsoft Academic Search returns a mere 19 publications for "Voting Feature Intervals" as compared with 1515 for Naïve Bayes and 24,255 for "Support Vector Machines")

VFI is similar to Naïve Bayes in that they both consider each feature separately. In VFI, each class is represented by a set of feature intervals on each feature dimension separately. Features participate in the classification by "voting", that is, each feature disperses real-valued votes among classes. The class receiving the highest number of votes is the assigned class of the document.

VFI has been shown to outperform Naïve Bayes and Classification by Feature Partition (CFP) both in speed and accuracy for most data sets in [26].

6.4 Hyperpipes

Hyperpipes [27] is a simple, fast classification algorithm. Similar to VFI, the algorithm relies on value intervals. However, in Hyperpipes the feature intervals are constructed for each class, as opposed to VFI where intervals are class independent. Classification using Hyperpipes is done by selecting the class with the most matching intervals.

7. EVALUATION

This work experiments with various preprocessing methods, feature selection methods and machine learning algorithms. It also experiments with several software tools to perform the tasks mentioned earlier. Different tools were utilized and tested for the preprocessing task. Alkhalil was used for stemming and rooting, Khoja for rooting, Stanford POS tagger [28] and Alkhalil to extract words by their tags and WEKA [29] to apply the machine learning algorithms. All experiments were performed on a collection of 416 poems, each 104 of which belonged to the same class.

Counter-intuitively, intricate preprocessing methods such as stemming and rooting did not increase classification accuracy. Quite the contrary, stemming decreased the accuracy slightly and rooting decreased it significantly. We attribute this decrease in accuracy to the relative inaccuracy of the stemmers and rooters used. The errors could also be attributed to the very nature of the Arabic language, as the semantic information in Arabic words lie not only in their roots, but also in the pattern the word assumes.

Extracting words by their POS tags did not increase classification accuracy as in [19]. Different algorithms were applied on nouns extracted via the Stanford POS tagger, verbs extracted via Stanford POS and verbs extracted via Alkhalil morphological analyzer. This is partially attributed to the nature of Arabic where adjectives are underutilized comparing to English. Also, the inaccuracy of the tools at hand could have played a role in this outcome.

As to feature selection, the mutual deduction method outperformed occurrence-count based selection. The length of the feature vector was parameterized from 400 to 2000 in steps of size 400.

Two SVM libraries were tested, LibLinear[30] and LibSVM[31] and LibLinear outperformed LibSVM significantly in all tests.

To test the results, standard 10 fold cross-validation with the standard precision, recall and F-measure measures from the Information Retrieval literature. Table II and (5), (6) and (7) define those measures. Multi-class precision-recall are calculated by calculating the precision and recall measures for each individual class then averaging over all four classes.

Both the LibSVM and the LibLinear SVM classifiers use four one-vs-all classifiers in order to extend classification to multi-class classification.

Table 2.Definition of testing terms

	Actual Class	
Predicted Class	tp (true positive)	fp (false positive)
	fn (false negative)	tn (true negative)

$$precision = \frac{tp}{tp+fp} \quad (5) \qquad recall = \frac{tp}{tp+fn} \quad (6)$$

$$F - measure = 2 * precision * \frac{recall}{precision+recall} (7)$$

As obvious from Table IV, the highest precision of 79% was achieved with feature vectors of length 800 and 2000 using the Hyperpipes algorithm. However, the highest F-measure, reaching up to 73% was achieved with a feature vector of length 2000 using VFI and both results where on non-stemmed, non-rooted, mutually deducted unigrams. This result however, cannot be accurately compared with the results in [12] due to the difference in training data and the unavailability of more comprehensive results, since only precision is listed in [12], as opposed to precision and recall.

An increase in both precision and recall is observed as the feature vector increases in size. This is similar to the result found in [22].

In terms of precision, the algorithms ranked as follows: $Hyperpipes > VFI > Naïve\ Bayes > SVM.$

Not all the results were listed due to size constraints. All the following results are for non-stemmed, non-rooted mutually deducted features.

Table 3.Results for classification using VFI on non-stemmed, non-rooted mutually deducted data

Voting Feature Intervals			
number of unigrams	**precision**	**recall**	F-**measure**
400	0.675	0.642	0.64
800	0.719	0.695	0.692
1200	0.75	0.730	0.730
1600	0.742	0.724	0.726
2000	**0.743**	**0.733**	**0.735**

Table 4.Results for classification using hp on non-stemmed, non-rooted mutually deducted data

HyperPipes			
number of unigrams	**precision**	**recall**	**F-measure**
400	0.749	0.45	0.419
800	0.792	0.582	0.594
1200	0.789	0.625	0.636
1600	0.783	0.642	0.651
2000	**0.791**	**0.654**	**0.659**

Table 5.Results for classification using Naïve Bayes on non-stemmed, non-rooted mutually deducted data

Naïve Bayes			
number of unigrams	**precision**	**recall**	**F-measure**
400	0.637	0.591	0.591
800	0.71	0.651	0.651
1200	0.732	0.663	0.662
1600	0.73	0.647	0.649
2000	0.736	0.659	0.661

Table 6.Results for classification using Liblinear (SVM) on non-stemmed, non-rooted mutually deducted data

LibLinear (SVM)			
number of unigrams	**precision**	**recall**	**F-measure**
400	0.564	0.567	0.564
800	0.593	0.594	0.592
1200	0.566	0.56	0.567
1600	0.571	0.575	0.572
2000	0.585	0.589	0.585

Table 7. Results for classification using Libsvm on non-stemmed, non-rooted mutually deducted data

LibSVM			
number of unigrams	**precision**	**recall**	**F-measure**
400	0.519	0.303	0.261
800	0.536	0.308	0.258
1200	0.528	0.305	0.263
1600	0.536	0.308	0.258
2000	0.527	0.305	0.251

8. CONCLUSIONS

This paper tested the susceptibility of Arabic poetry to emotion classification. The problem was treated as a text categorization problem, classifying poems into four classes: Retha, Ghazal, Heja and Fakhr. Various preprocessing methods we tested, feature selection methods and machine learning algorithms. Finally, four main machine learning algorithms are compared: Naïve Bayes, SVMs, VFI and Hyperpipes. The best precision achieved was 79% using Hyperpipes with non-stemmed, non-rooted, mutually deducted feature vectors containing 2000 features.

9. ACKNOWLEDGEMENT

We would like to thank JadDarrous who implemented the screen scraper to get the poems.

10. REFRENCES

[1] C.I. Nass, J.S. Stener, E. Tanber, “Computers are social actors,” in Proc. of the SIGCHI conference on Human factors in computing systems: celebrating interdependence, Boston, MA, April, 1994, pp. 72-78.

[2] J. Bates. (1997, July). The Role of Emotion in Believable Agents. Communications of the ACM [online]. 37(7), pp. 122-125. Available: http://dl.acm.org/citation.cfm?id=176803.

[3] R. W. Picard, Affective Computing, Cambridge, MA: MIT Press, 1997.

[4] T. S. Polzin, A. Waibel, “Emotion-Sensitive Human-Computer Interfaces,” presented at the ISCA Workshop on Speech and Emotion, Northern Ireland, 2000.

[5] R. Fernandez, "A Computational Model for the Automatic Recognition of Affect in Speech," Ph.D. thesis, MIT Media Arts and Science, MIT, Cambridge, MA, 2004.

[6] J. Riseberg, J. Klein, R. Fernandez, R.W. Picard, "Frustrating the User on Purpose: Using Biosignals in a Pilot Study to Detect the User's Emotional State," CHI late-breaking, March 1998.

[7] G. Mishne, “Experiments with Mood Classification in Blog Posts,” in Proc. of the 1st Workshop on Stylistic Analysis of Text for Information Access, Salvador, Bahia, Brazil, 2005.

[8] L. Holzman, W. Pottenger, “Classification of emotions in internet chat: An application of machine learning using speech phonemes,” Comp. Sc. Eng., Lehigh Uni., Jackson, NJ, LU-CSE-03-002, 2003.

[9] F. Keshtkar, D. Inkpen. “Using Sentiment Orientation Features for Mood Classification in Blog Corpus,” IEEE International Conference on Natural Language Processing and Knowledge Eng., Dalian, China, Sep., 2009.

[10] H. Liu, H. Lieberman, T. Selker, “A Model of Textual Affect Sensing using Real-World Knowledge,” in Proc. of the Seventh International Conference on Intelligent User Interfaces, Miami Beach, Florida, 2003 , pp. 125-132.

[11] A. C. Boucouvalas, X. Zhe, “Real Time Text-To-Emotion Engine for Expressive Internet Communications,” presented at 8th International Symposium on Communication Systems, Networks & Digital Signal Processing, Poznań , Poland, 2002.

[12] S. Al-Harbi, A. Almuhareb, A. Al-Thubaity, M. S. Khorsheed, A. Al-Rajeh, “Automatic Arabic Text Classification,” presented at the 9th International Conference on the Statistical Analysis of Textual Data, Lyon, France, 2008.

[13] A. Abbasi, H. Chen, A. Salem. (June, 2008). Sentiment analysis in multiple languages: Feature selection for opinion classification in web forums. [online]. ACM Trans. Inf. Syst. 26(3). Available: http://dl.acm.org/citation.cfm?id=1361685&dl=ACM&coll=DL&CFID=84879471&CFTOKEN=15130439.

[14] M. A. Mageed, M. T. Diab, M. Korayem, “Subjectivity and Sentiment Analysis of Modern Standard Arabic,” presented at the 49th Annual Meeting on Association for Computational Linguistics. Portland, Oregon, USA, June, 2011.

[15] A. Ortony, G. L. Clore, A. Collins, The cognitive structure of emotions, New York: Cambridge University Press. 1988.

[16] C. Elliott, "The Affective Reasoner: A Process Model of Emotions in a Multi-agent System," Ph.D. thesis, Inst. Learning Sciences, Northwestern Univ.,Tech. Rep. 32, 1992.

[17] M.G. Dyer. (1987). Emotions and Their Computations: Three Computer Models. Cognition and Emotion. 1(3), 323-347.

[18] Y. Jung, H. Park, and S. Myaeng, "A hybrid mood classification approach for blog text," in proc. of the 9th Pacific Rim international conference on Artificial intelligence, Berlin, Germany, 2006, pp. 1099–1103.

[19] O. Alsharif, D. Alshamaa, J. Darrous. N.ghneim "Opinion Mining", Internalreport, (in Arabic), Artificial Intelligence Department, Information Technology Engineering Faculty, Damascus University, Syria, 2011.

[20] Alkhalil Morphological Parser. Internet: http://alkhalil.sourceforge.net/ Jan. 10, 2012 [Jan. 10, 2012]

[21] ShereenKhoja. "Khoja Arabic Stemmer". Internet: http://zeus.cs.pacificu.edu/shereen/research.htm. Aug. 16th, 2010 [Jan. 10, 2012].

[22] B. Pang, L. Lee, S. Vaithyanathan, "Thumbs up? Sentiment Classification using Machine Learning Techniques," in the Conference on Empirical Methods in Natural Language Processing, PA,2002, pp.79-86.

[23] D. D. Lewis, "Naive (Bayes) at forty: The Independence Assumption in Information Retrieval," in Proc. of the European Conference on Machine Learning, 1998, pp. 4–15.

[24] T. Joachims, "Text Categorization with Support Vector Machines: Learning with Many Relevant Features." Presented at the European Conference on Machine Learning, Chemnitz, Germany, 1998.

[25] Y. Yang, X. Liu., "A Re-examination of Text Categorization Methods," in Proc. of the 22nd Annual International ACM SIGIR Conference on Research and development in Information Retrieval, New York, NY, 1999, pp. 42–49.

[26] G. Demiröz and H. Güvenir, "Classification by Voting Feature Intervals," presented at the 9th European Conference on Machine Learning, London, UK, 1997, pp. 85–92.

[27] S. E. Hansen., "Solving Classification Problems through Automatic Programming," M.S. thesis., Dep. Comp. Sc. ,Østfold Univ. Col., Halden, Norway, 2007.

[28] Kristina Toutanova, Dan Klein, Christopher Manning, and Yoram Singer. 2003. Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network.

[29] I.H. Witten, E. Frank, Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations. Morgan Kaufmann, San Francisco, CA. Available on WWW: www.cs.waikato.ac.nz/ml/weka

[30] R.E. Fan, K.W. Chang, C. J. Hsieh, X.R. Wang, C.J. Lin. LIBLINEAR: A library for large linear classification Journal of Machine Learning Research 9(2008), 1871-1874.

[31] C. C. Chang, C. J. Lin, LIBSVM: A Library for Support Vector Machines. ACM Transactions on Intelligent Systems and Technology, 2:27:1--27:27, 2011. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.